

File Encryption and Authentication Utility

Course Project for CS681 & 392
Fall 2004

Convenience of booting a machine with a floppy disk (or CD) can bypass any security mechanism built into an operating system. This is inevitable because if the operating system is not active, it cannot protect any data. This means anyone who has physical access to a system can steal data from that system by just rebooting a machine with a different operating system. One of the solutions to address this problem is to have a file system that can protect the data in it.

This semester you will design and implement a file encryption and authentication utility (FEAU, pronounced “pheew”) to address the aforementioned problem. The design and implementation will be carried out in several stages as you progress through the semester by solving a specific problem in each stage. Note that, designing and implementing a utility in stages requires that the overall design of the utility to be modular. Our goal is to learn the intricacies in designing a secure system first hand and coming up with systematic methods to handle the challenges. The rest of the document lays out the specifications, requirements, and a tentative development schedule for the project. If you have questions please contact either Vikram or Kulesh.

1 FEAU Specification

FEAU is a standard shell for *nix based operating systems. It provides all the functionalities of a regular shell and some extended functionalities that are unique to FEAU. The extended functionalities include:

- protecting user files from unauthorized access (even when FEAU is disabled)
- maintain file integrity (even when file is modified outside of FEAU)
- accountability (or non-repudiation) in a multi-user environment

Now we describe all functionalities and requirements of FEAU in detail.

1.1 Overview of Functionalities

1. **File Manager:** FEAU provides functionalities of a simple file manager such as listing, copying, renaming, and moving files. **Design Hint:** *Note that we may choose to add more commands at a later time. Therefore, your design must be modular and should easily accommodate future commands.*
2. **User and Group Management:** FEAU provides authorization, authentication, and accounting to users and groups. FEAU’s user and group management system is independent of that of the host operating system.
3. **File Encryption and Decryption:** FEAU protects user files by encrypting it prior to storing it in the file system and decrypting it upon user access. **Design Hint:** *Note that multiple users (in different groups) may share a file. Also, a group may share a file. Your design should accommodate such special cases.*
4. **File Authentication and Integrity Checking:** FEAU allows users to sign both files managed by FEAU and some arbitrary file in the file system. FEAU should automatically check for file signature, if found it should verify the integrity of the file and inform the user whether the file is verified or not.

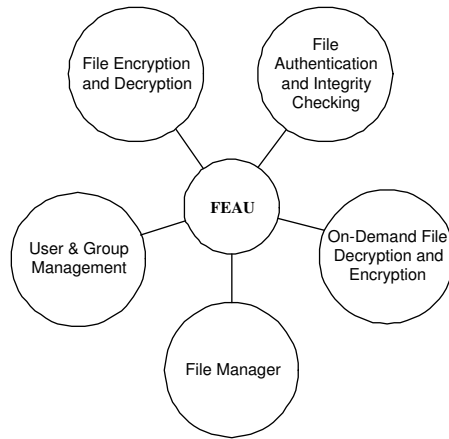


Figure 1: FEAU Functionalities

5. **Transparent Operations:** FEAU's operations must be transparent to the end-user. That is, for example, if an user choose to edit an encrypted file by entering the command "`vi feau.c`" then FEAU should first decrypt the file, send the file to the `vi` editor, and when `vi` editing session is over it should encrypt the new file and store it in the file system.

1.2 Requirements

Your FEAU implementation SHOULD provide the following:

1. An interactive environment for users similar to UNIX shell and should support both internal command and external commands. (See Section 3)
2. Secure any file type (text, binary, executable etc.)
3. Allow multi user/group encrypted file sharing. Users should be able to specify who else or which group(s) of users can decrypt their files.
4. Capable of handling more than one administrative user.
5. Allow administrative users to decrypt any file.
6. Allow administrative user to add, delete, and modify users and groups.
7. Allow administrative users to specify and modify security policies.
8. Maintain a log of all activities for accounting and non-repudiation.
9. Use `*nix` password file format to store user passwords, and your password storage mechanism should be available to public.
10. Must work in `*nix` environment

2 Development Schedule

As stated earlier, you will be developing FEAU in stages through out the semester. Here is a tentative schedule for assignments and the description of the problem you will be addressing in each assignment.

1. **Introduction to C and FEAU design** In this assignment you will learn some concepts of C language that will be useful to you for the development of FEAU. You will also design FEAU's overall architecture and a simple file manager module. You will also implement and test the simple file manager.
2. **Private Key Cryptography** In this assignment you will learn some properties of AES and design and implement the file encryption module for FEAU.
3. **Public Key Cryptography** In this assignment you will learn some properties of RSA and design and implement the user and group key management module.

4. **Hash Algorithms** In this assignment you will learn some properties of MD5 hash algorithm and complete the design and implementation of key management module.
5. **Authentication, Authorization, Accounting** In this assignment you will learn about AAA mechanisms, design and implement AAA module for FEAU.
6. **Reference Monitor** In this assignment you will design and implement a reference monitor module that will enforce security policies in FEAU.
7. **Common Code Vulnerabilities** In this assignment you will learn about buffer overflows and format string vulnerabilities in detail. Then you will analyze colleagues' FEAU implementation for these vulnerabilities and write proof-of-concept code to exploit one of the vulnerability you found.
8. **Robust Programming** In this assignment you will learn about robust programming techniques, analyze your FEAU, and recode to make it robust. You will also fix any vulnerability found in the previous assignment by your colleagues.
9. **Password Auditing** In this assignment you will explore common password cracking techniques and crack FEAU password file of your colleague's FEAU implementation.
10. **Code Auditing** In this assignment you will audit your colleagues FEAU code and create attack trees.
11. **Linux and Windows Security** In this assignment you will evaluate the security of the Linux and Windows box given to you and give recommendations for securing them.

3 FEAU Internal Commands

Following are the internal commands that should be supported:

1. **ls - List** current directory contents

Synopsis - ls [-aAeEl] [file...]

Description For each operand that names a file of a type other than directory, ls displays its name as well as any requested, associated information. For each operand that names a file of type directory, ls displays the names of files contained within that directory, as well as any requested, associated information. The following option are available:

- a List all entries
- A List all entries including encrypted files
- e List encrypted files
- E List all encrypted files, that can be read by the current user.
- l List all information associated with a file.

2. **cp - Copy** files

Synopsis - cp [-e] [source file] [-e] [target file]

Description cp copies the contents of the source file to the target file. The following options are available:

- e indicates encrypted file, the file will be decrypted when used with source file, and will be encrypted when used with target file.

3. **mv - Move** files

Synopsis - mv [-e] [source file] [-e] [target file]

Description mv moves the contents of the source file to the target file. The following options are available:

-e indicates encrypted file, the file will be decrypted when used with source file, and will be encrypted when used with target file.

4. **lsusr - list** user or group

Synopsis - lsusr [-g] [group...] [-G]

Description lsusr lists all user by default. The following options are available:

- g lists only the member of group
- l list all information associated with a user or group
- G lists all groups

5. **addusr - Add** user

Synopsis - addusr [-u] or [-r] or [-g] [string...]

Description addusr adds a new user. The following options are available:

- u user name
- r users real name, this is optional. Default is null
- g indicate the group(s) to which this user will be added

6. **delusr - Delete** user

Synopsis - delusr [user1] [user2]

Description delusr deletes a user1, and if user2 is specified then encrypted file owned by user1 will be transferred to user2.

7. **modusr - Modifies** user

Synopsis - modusr [username] [-g] or [-G] [group]

Description modusr modifies a users group membership. The following options are available:

- g added user to the group
- G removes user from the group

8. **setpass - Password** set or reset

Synopsis - setpass [user] [password]

Description setpass set or reset user's password.

9. **addgrp - Add** group

Synopsis - addgrp [-u] or [-r] or [-g] [string...]

Description addusr adds a new group. The following options are available:

- u group name
- r group description, this is optional. Default is null
- g Existing users could become a member of this group, if specified here. This is optional, default is null.

10. **delgrp - Delete** group

Synopsis - delgrp [group1] [group2]

Description delgrp deletes a group1, and all member of group1 will be transferred to group2. If group2 is omitted, all users who are unique to group1 will be deleted.

11. **enc - Encrypt** files

Synopsis - enc [file...]

Description enc encrypts the contents of the file, delete the source file, and move the encrypted file to secured location. More then one file, or a whole directory could be encrypted.

12. **dec - Decrypt** files

Synopsis - dec [file...]

Description dec decrypts the contents of the file in the secured location, delete the source file, and move the decrypted file to its original location. More then one file, or a whole directory could be decrypted.

13. **fprm - File Permissions**

Synopsis - flattr [-a] or [-s][file...] [-rw] [user...] [group...]

Description flattr sets or resets file permissions. The following options are available

- a append to existing file permission, an error is returned if permission conflict.
- s replace existing file permission.
- r Allow read access
- w Allow write access

14. **snfl - Sign** files

Synopsis - snfl [file...]

Description snfl signs both encrypted and plain file with current users private key.

15. **vfy - Verify** file integrity

Synopsis - vfy [files...]

Description vfy check file integrity.

16. **rdplcy - Read Policy** file

Synopsis - rdplcy [file]

Description rdplcy reads security policies from an external file and replaces the current policies.

17. **svplcy - Save Policy** to afile

Synopsis - svplcy [file]

Description svplcy saves current security policies to a file.

18. **bksys - Backup**

Synopsis - bksys [file]

Description Backup all user and group information, security policies, and all other system information to a file.

19. **retsys - Restore**

Synopsis - retsys [file]

Description Restores all system information, user and group information, and security policies from a backup file.

20. **[external command] - External** command

Synopsis - [external command] [file...]

description - Any command that are not part of Internal commands are considered as external command and they are passed to the shell. If file name or names are specified, they will be decrypted and is passed to the external command and will be re-encrypted when the external command finishes.